IT IS CLAIMED AS THE INVENTION:

1. A computer-implemented B-tree structure for information processing involving a database system with a plurality of data records, wherein a set of the data records have duplicate keys, comprising:

a plurality of interconnected nodes having a root node, index nodes and leaf nodes;

wherein a leaf node is configured to store a first key corresponding to first data in a first data page;

wherein the first data in the first data page is configured to store a second key that is a duplicate of the first key and that corresponds to second data stored on a second data page.

2. The B-tree structure of claim 1 wherein said first data page and second data page comprise the same page.

3. The B-tree structure of claim 1 wherein said first data page and second data page comprise different pages.

4. The B-tree structure of claim 1 wherein said first data and second data are the same.

5. The B-tree structure of claim 1 wherein said first data and second data are different.

6. The B-tree structure of claim 1 wherein said first data has variable length.

7. The B-tree structure of claim 1 wherein said second data has variable length.

8. The B-tree structure of claim 7 wherein degree of the leaf nodes is not substantially affected by the variable length of the first and second data.

9. The B-tree structure of claim 8 wherein degree of the leaf nodes is not substantially affected because the first and second data are stored separate from the leaf nodes.

10. The B-tree structure of claim 1 wherein said plurality of leaf nodes are maintained in sequential order and with a doubly linked list which connects each of said leaf node with its sibling nodes.

11. The B-tree structure of claim 10 wherein the B-tree is configured to operate with a find operation.

12. The B-tree structure of claim 10 wherein the B-tree is configured to operate with a find-next operation.

13. The B-tree structure of claim 10 wherein the B-tree is configured to operate with a find-previous operation.

14. The B-tree structure of claim 10 wherein the B-tree is configured to operate with a find-first operation.

15. The B-tree structure of claim 10 wherein the B-tree is configured to operate with a find-last operation.

16. The B-tree structure of claim 10 wherein the B-tree is configured to operate with an insert operation.

17. The B-tree structure of claim 10 wherein the B-tree is configured to operate with a delete operation.

18. The B-tree structure of claim 1 wherein data associated with the first and second keys are stored separate from the leaf nodes.

19. The B-tree structure of claim 1 wherein the first and second keys each have a corresponding unique data record value.

20. The B-tree structure of claim 1 wherein substantially concurrently executing processes update the first and second keys at approximately the same time without being locked out by another process because the first and second data are stored on different data pages.

21. The B-tree structure of claim 20 wherein the processes are threads.

22. The B-tree structure of claim 1 wherein page and offset for the second key's value follow the second data on the second data page.

23. The B-tree structure of claim 1 wherein each page has associated with it a lock handle, wherein because the B-tree is self-balancing, an insert operation to the B-tree avoids locking the entire B-tree or subtree.

24. The B-tree structure of claim 1 wherein the leaf nodes contain more than two key-value entries.

25. The B-tree structure of claim 1 wherein the second key is a duplicate key of the first key, wherein the second data is configured to store a third key that is a duplicate of the first key and that corresponds to third data stored on a third data page.

26. The B-tree structure of claim 1 wherein the second key is a duplicate key of the first key, wherein the second data is configured to store a third key that is a duplicate of the first key and that corresponds to third data stored on the second data page.

27. A computer-implemented method for concurrent execution of a plurality of transactions in a database system containing a plurality of data records, wherein a set of the data records have duplicate keys, said method comprising:

storing said plurality of data records in a B* tree structure with a plurality of index nodes and a plurality of leaf nodes, wherein each of said leaf nodes includes a plurality of elements each having a first pointer configured to store a first key corresponding to first data in a first data page;

wherein said first data further includes a second pointer configured to store a second key that is same as said first key and that corresponds to second data in a second data page;

implementing said plurality of transactions by concurrently locating and operating on the target data records stored in said data pages through use of said B* tree structure.

28. The method of claim 27 wherein said step of implementing said plurality of transactions further includes implementing a concurrency control protocol.

29. The method of claim 28 wherein the concurrency control protocol controls a first of said transactions to access first data in the first data page and concurrently a second of said transactions to access second data in the second data page, wherein said first data and second data have the same key.

30. The method of claim 28 wherein the concurrency control protocol is a lock-based protocol.

31. The method of claim 28 wherein the lock-based protocol releases locks on index nodes and leaf nodes when the data page is identified.

32. A computer-readable medium for concurrent execution of a plurality of transactions in a database system containing a plurality of data records, wherein a set of the data records have duplicate keys, comprising instructions for:

storing said plurality of data records within a B* tree structure that has a plurality of index nodes and a plurality of leaf nodes, wherein each of said leaf nodes includes a plurality of elements having a first pointer configured to store a first key corresponding to first data in a first data page;

wherein said first data further includes a second pointer configured to store a second key that is same as said first key and that corresponds to second data in a second data page;

implementing said plurality of transactions by concurrently locating and operating on the target data records stored in said data pages.

33. An information processing system in database application, comprising:

a plurality of data records with a first set of data records having duplicate keys, said plurality of data records stored in a B* tree structure with a plurality of index nodes and a plurality of leaf nodes, wherein each of said leaf nodes includes a plurality of elements having a first pointer configured to store a first key which corresponds to first data stored in a first data page;

wherein said first data includes a second pointer configured to store a second key that is a duplicate of the first key and that corresponds to second data in a second data page;

an engine for implementing a plurality of transactions by concurrently locating and operating on the data records stored in the data pages;

a concurrency-control manager for implementing a concurrency control protocol through use of the B* tree structure.